CSS Grid

Agenda

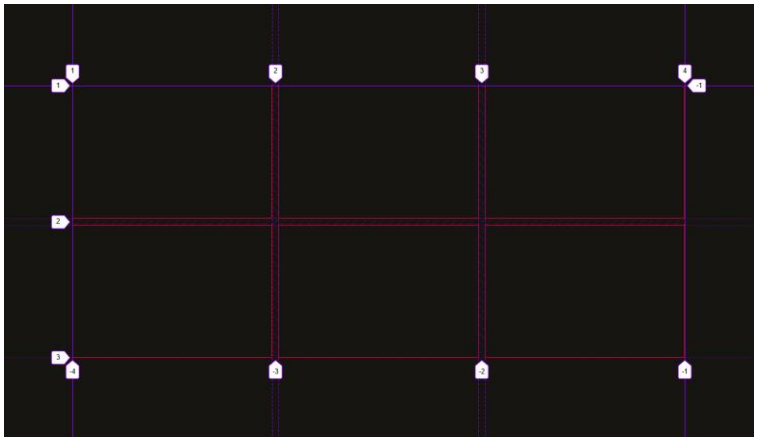
¿Qué es CSS Grid?

Sistema de maquetación en rejilla pensado para distribuir el contenido en filas y/o columnas

Permite trabajar sobre dos ejes, el horizontal (x) y el vertical (y)

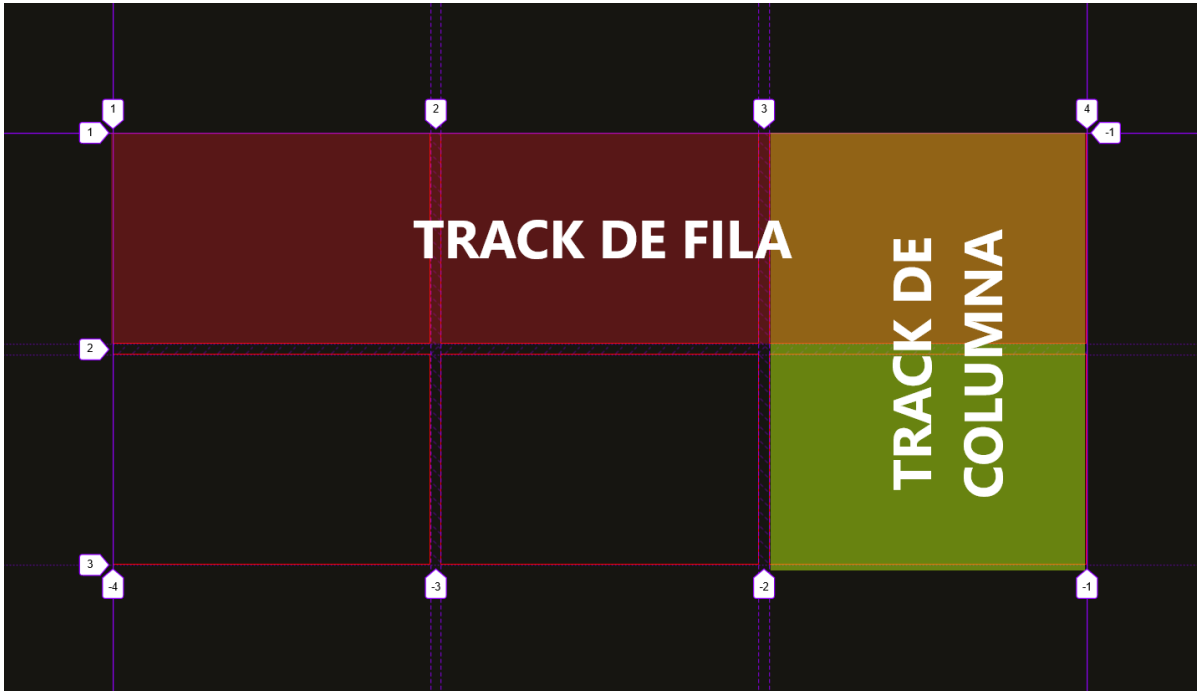
Permite dividir una página o elemento en áreas o regiones

Se define en la propiedad `display` con el valor `display: grid` o `display: inline-grid`



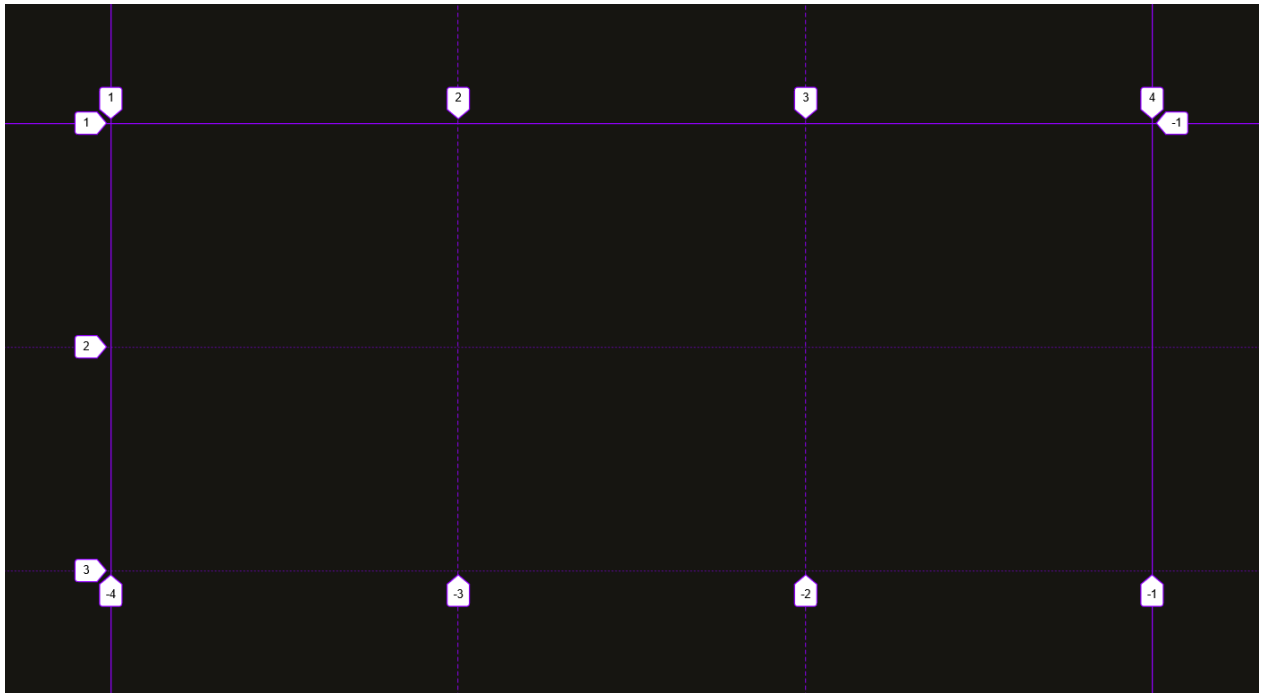
Grid Tracks o pistas

Grid **Tracks** o **pistas** es el espacio entre dos líneas. Definen la fila (grid row) o la columna (grid column).



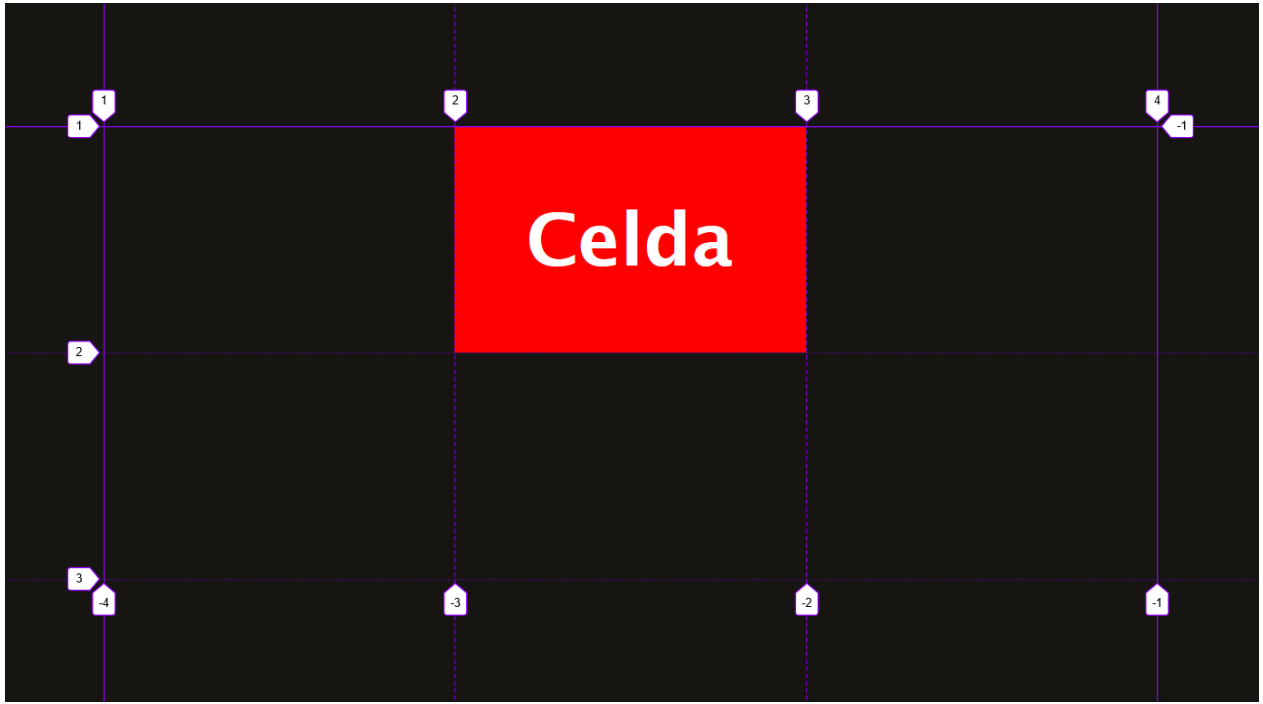
Líneas de cuadrícula

Se crean al definir las pistas. La numeración de la imagen muestra el número de línea vertical y horizontal.



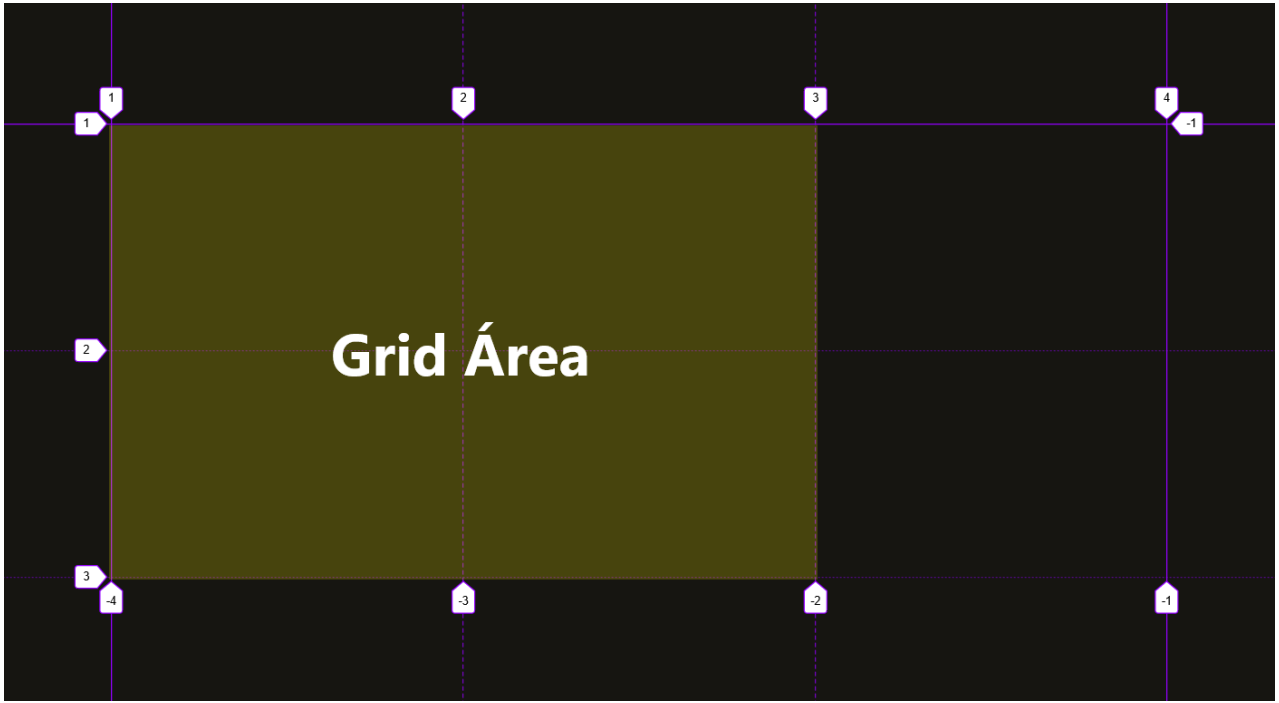
Grid Cell o celda

Grid **Cell** o **celda** es la división más pequeña que puede tener un grid



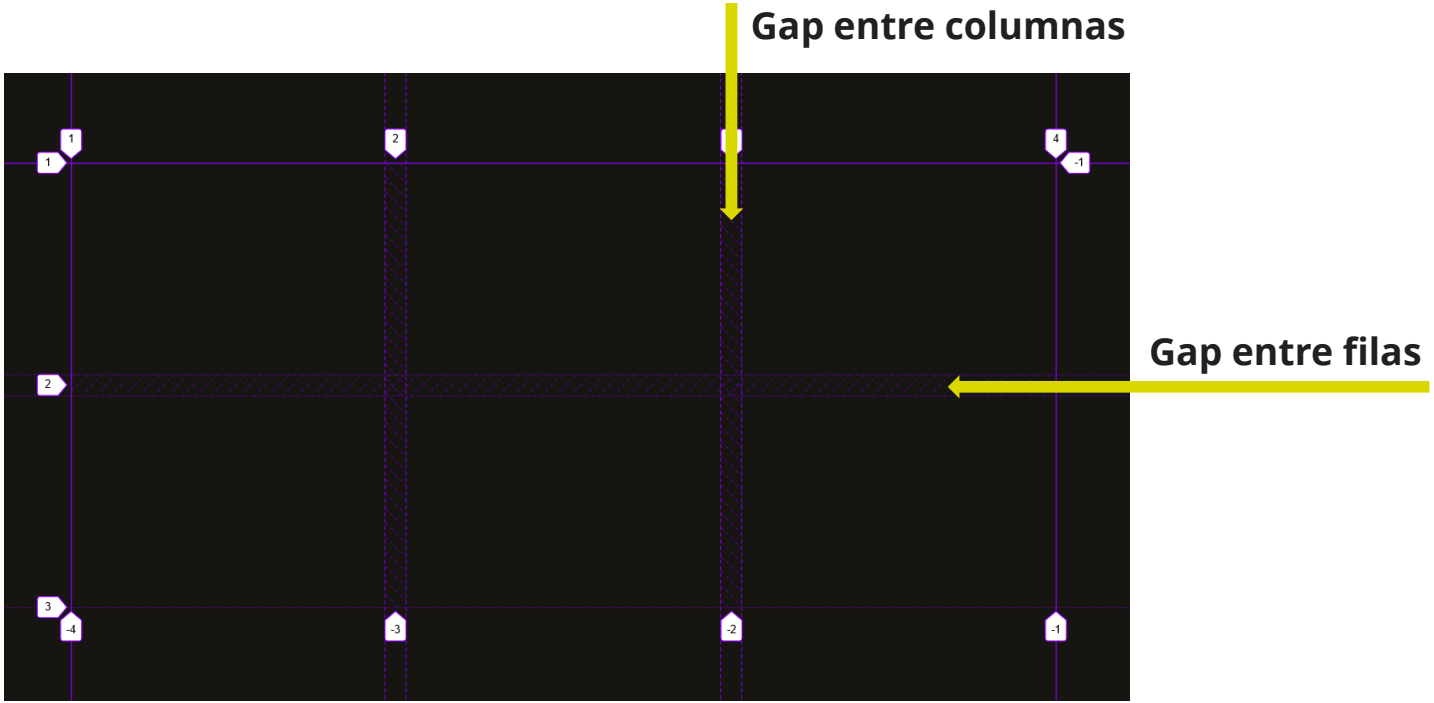
Grid Áreas

Grid Áreas se crean a partir de una o más celdas y siempre deben ser de naturaleza rectangular



Grid Gutters o Gap

Grid Gutters o **Gap** es un espacio que podemos controlar y que separa cada pista o track



Antes de empezar...

Cuando hablamos de **cuadrícula explícita** nos referimos a las filas y columnas que definimos mediante alguna de las propiedades que veremos a continuación

La **cuadrícula implícita** hace referencia a las filas y columnas no definidas que grid puede crear si fuera necesario para añadir contenido. Este comportamiento también podremos controlarlo

Propiedades para el control de estructura

grid-template-columns define el ancho de una o varias columnas de manera individual pudiendo asignar un nombre a cada una de las líneas (verticales)

```
grid-template-columns: [primera-linea] 50px [segunda-linea] 20% 1fr auto [ultima-linea];
```

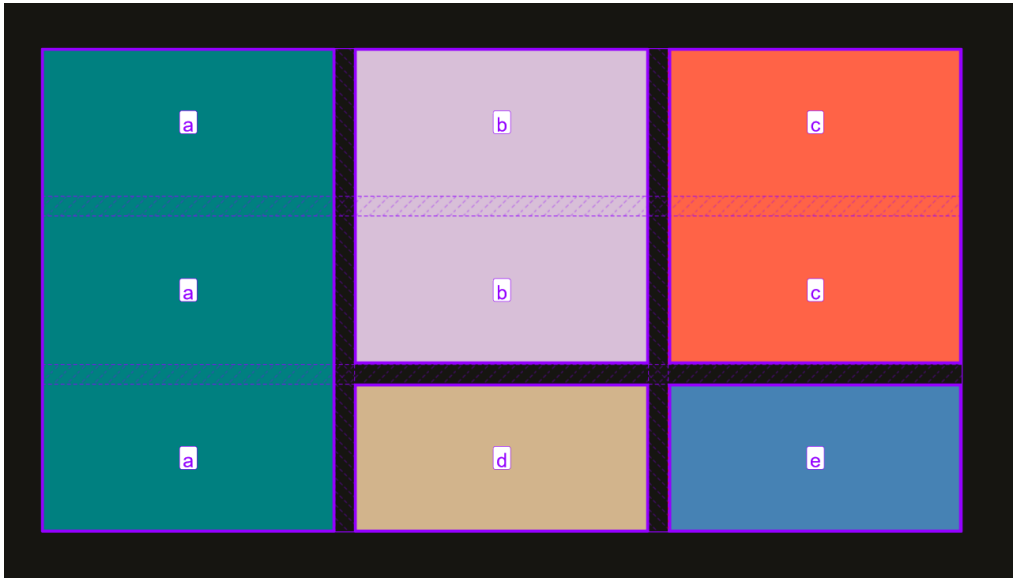
grid-template-rows define el alto de una o varias filas de manera individual pudiendo asignar un nombre a cada una de las líneas (horizontales)

```
grid-template-rows: 100px 100px 200px;
```

Propiedades para el control de estructura II

grid-template-areas define y da nombre a cada uno de los **grid area**

```
grid-template-areas:
    "a b c"
    "a b c"
    "a d e";
```



Propiedades para el control de estructura III

grid-template es una propiedad que engloba las tres anteriores (shorthand para area, row y column)

```
grid-template:
    [f1L1]"a b c" 40px [f1L2] /* primera fila */
    "a b c" 40px /* segunda fila */
    "a d e" 40px /* tercera fila */
    / [c1L1] 1fr [c1L2] 1fr 1fr; /* estructura de columna */
```

¡¡Consejo!! intenta evitar utilizar este tipo de propiedades abreviadas ya que puede resultar confuso y es más fácil cometer errores.

Propiedades para el control de estructura IV

grid-auto-columns define el ancho de todas las posibles columnas no definidas en el Grid Container

```
grid-auto-columns: 200px;
```

grid-auto-rows define el alto de todas las posibles filas no definidas en el Grid Container

```
grid-auto-rows: auto;
```

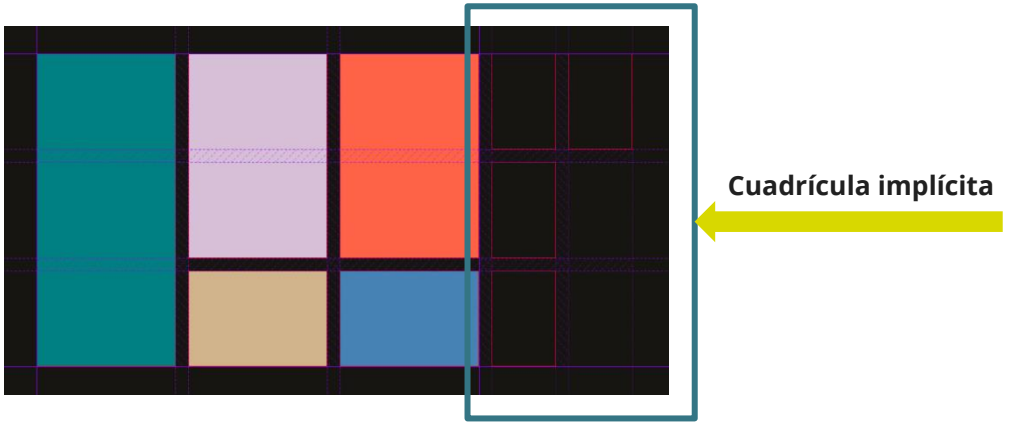
Con estas propiedades controlamos la **cuadrícula implícita**

Propiedades para el control de estructura V

grid-auto-flow Define la dirección en donde se ubicaran los elemento que no estén definidos como filas y columnas dentro del Grid Container

```
grid-auto-flow: column;
```

```
grid-auto-columns: 100px;
grid-auto-flow: column;
```

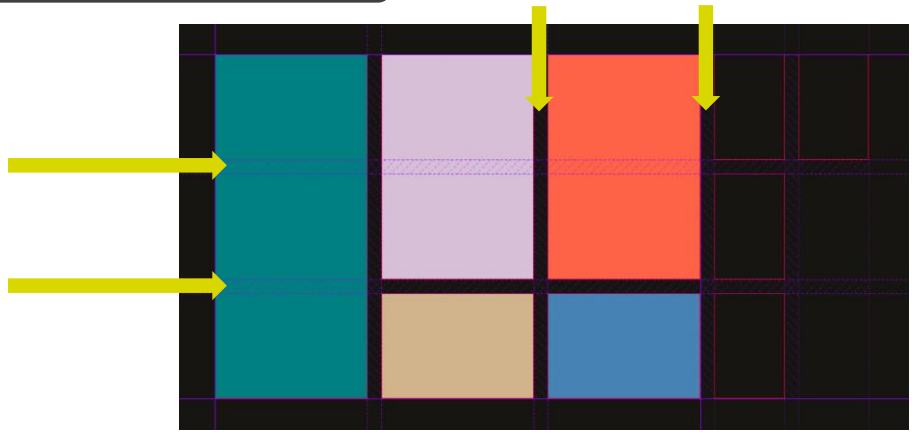


Con estas propiedades controlamos la **cuadrícula implícita**

Propiedades de espaciado

row-gap, **column-gap** y **gap** definen el gutter o espaciado entre columnas y filas

`row-gap: 10px;`	establece la separación entre filas
`column-gap: 10px;`	establece la separación entre columnas
`gap: 10px 10px;`	**Shorthand**. Establece la separación entre filas y columnas, si sólo se proporciona un valor este se aplica para ambas



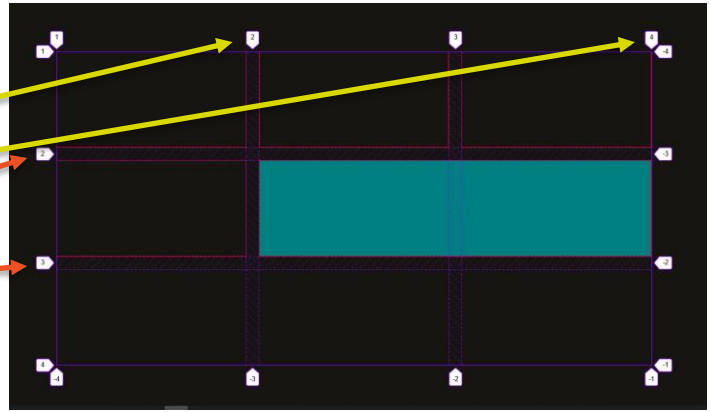
Posicionar elementos respecto a líneas

```
grid-row-start || grid-row-end || grid-column-start || grid-column-end
```

Se pueden usar de manera individual o conjunta para definir el inicio y fin del elemento en filas y columnas

Si las columnas y filas se han definido nombradas podemos utilizar el nombre en vez del número

```
.element {
  grid-column-start: 2; /* define el inicio desde la línea de columna indica */
  grid-column-end: 4; /* define el fin en la línea de columna indicada */
  grid-row-start: 2; /* define el inicio desde la línea de fila indicada */
  grid-row-end: 3; /* define el fin en la línea de fila indicada */
}
```



Posicionar elementos respecto a líneas II

`grid-row || grid-column`

Son los shorthand para poder definir principio y fin en una sola propiedad

```
.element {
  grid-column: 2 / 4; /* define el inicio y fin respecto a columnas*/
  grid-row: 2 / 3; /* define el inicio y  fin respecto a filas*/
}
```

Otros valores que podemos utilizar:

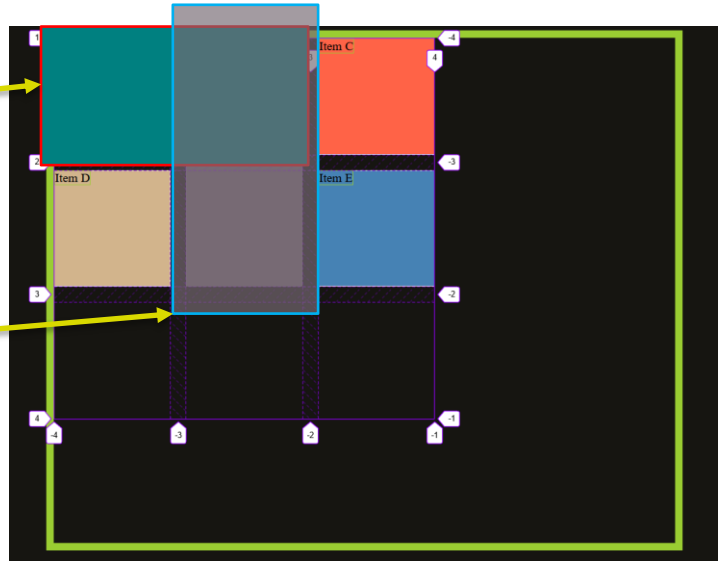
`grid-row: 1 / -1;`	**-1** como valor de fin de fila o columna indica que ocupe todo el espacio de fila o columna
`grid-column: 1 / span 4;`	**span** (se extiende) el elemento ocupará el número de filas o columnas indicado, en este caso empieza en la línea 1 de columnas y se extiende 4 más después de la 1ª.
`grid-row: 1 / auto;`	**auto** define el inicio o final de una fila o columna de manera automática

Superposición de elementos

Los elementos de un grid pueden ocupar una misma celda superponiéndose entre ellos

Podemos controlar este orden de superposición con la propiedad **z-index** en cada elemento

```
.itemA {
  grid-column: 1 / 3;
  grid-row: 1 / 2;
  z-index: -100;
}
.itemB {
  grid-column: 2 / 3;
  grid-row: 1 / 3;
  opacity: 0.5;
}
```



Funciones

repeat()

- Usada en las propiedades **grid-template-columns** y **grid-template-rows**
- Define un número repetido de filas o columnas
- Admite dos parámetros, número de repeticiones y tamaño
- grid-template-columns: repeat(4, 100px);

minmax()

- Define un rango de tamaño
- Mayor o igual que min. y menor o igual que max.
- Usada en **grid-template-columns**, **grid-template-rows**, **grid-auto-columns**, **grid-auto-rows**
- grid-template-columns: minmax(50px, 100px) 1fr;

Valores

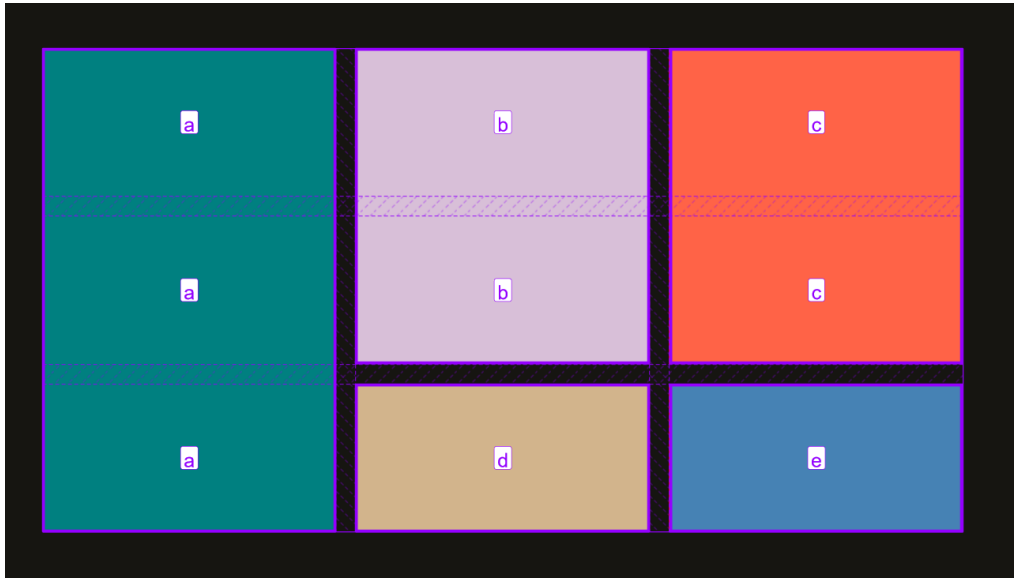
min-content	Define el tamaño más pequeño de una caja sin que el contenido llegue a desbordar
max-content	Define el tamaño máximo en función de su contenido, minimiza el espacio vacío y evita el desbordamiento
fr	Representa una fracción del espacio disponible en el grid container

Podemos utilizar estos valores en las funciones anteriores para definir tamaños en filas y columnas

Posicionar elementos con grid area

grid-area se aplica a un elemento para indicar qué área definida debe ocupar

```
.grid-container {
    grid-template-areas:
        "a b c"
        "a b c"
        "a d e";
}
.itemA {
    grid-area: a;
}
.itemB {
    grid-area: b;
}
```



Alineación de elementos en un grid

```
justify-content || align-content
```

Permiten alinear y distribuir las columnas y filas de manera horizontal (justify-content) y vertical (align-content) dentro del grid container

flex-start posiciona la cuadrícula al inicio de su espacio vertical y/o horizontal (valor por defecto)

```
justify-content: flex-start;
align-content: flex-start;
```

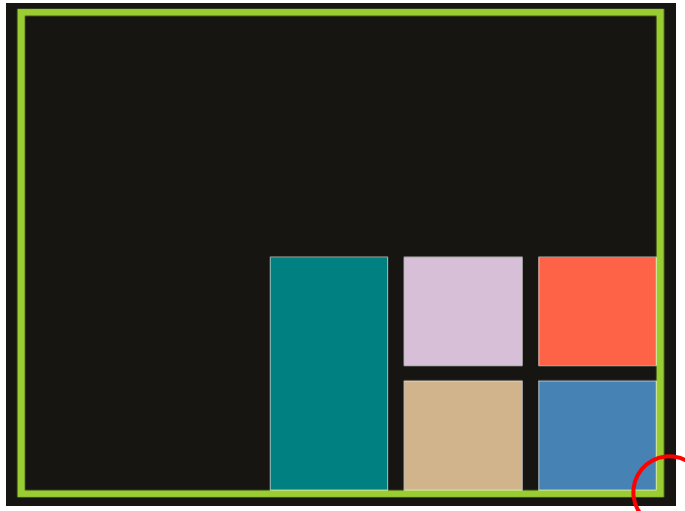
Alineación de elementos en un grid

```
justify-content || align-content
```

Permiten alinear y distribuir las columnas y filas de manera horizontal (justify-content) y vertical (align-content) dentro del grid container

flex-end posiciona la cuadrícula al final de su espacio vertical y/o horizontal

```
justify-content: flex-end;
align-content: flex-end;
```



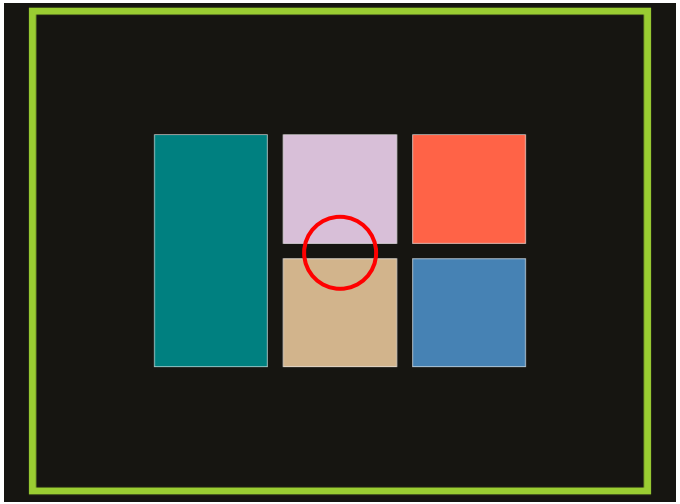
Alineación de elementos en un grid

```
justify-content || align-content
```

Permiten alinear y distribuir las columnas y filas de manera horizontal (justify-content) y vertical (align-content) dentro del grid container

center → posiciona la cuadrícula en el centro de su espacio vertical y/o horizontal

```
justify-content: center;
align-content: center;
```



Alineación de elementos en un grid

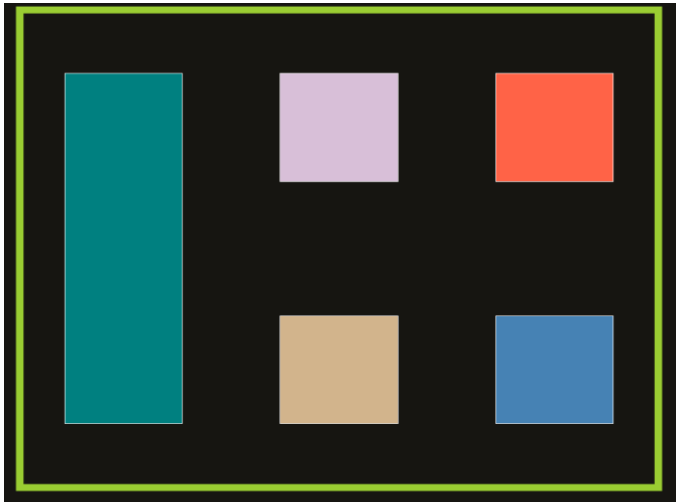
```
justify-content || align-content
```

Permiten alinear y distribuir las columnas y filas de manera horizontal (justify-content) y vertical (align-content) dentro del grid container

space-around

```
justify-content: space-around;
align-content: space-around;
```

Distribuye filas y columnas en el espacio disponible vertical y/o horizontal dejando el mismo espaciado entre ellos y otro variable alrededor



Alineación de elementos en un grid

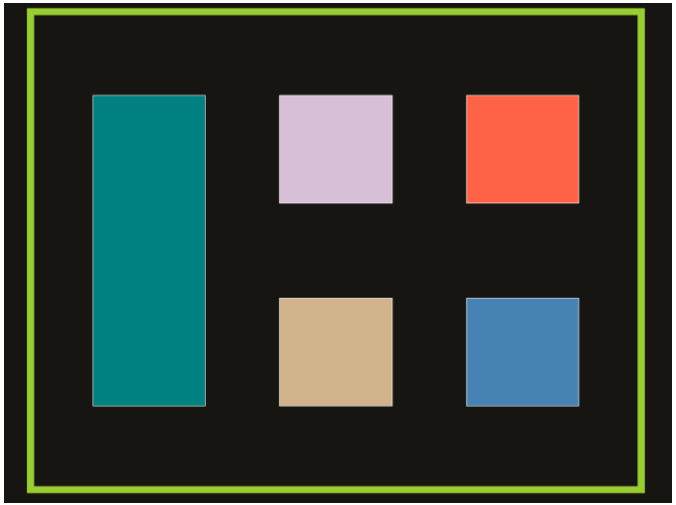
```
justify-content || align-content
```

Permiten alinear y distribuir las columnas y filas de manera horizontal (justify-content) y vertical (align-content) dentro del grid container

space-evenly

```
justify-content: space-evenly;
align-content: space-evenly;
```

Distribuye filas y columnas en el espacio disponible vertical y/o horizontal dejando el mismo espaciado entre y alrededor de ellos



Alineación de elementos en un grid

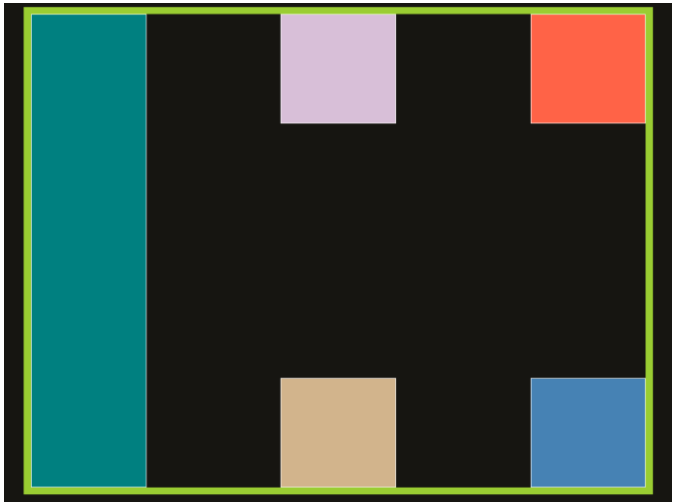
```
justify-content || align-content
```

Permiten alinear y distribuir las columnas y filas de manera horizontal (justify-content) y vertical (align-content) dentro del grid container

space-between

```
justify-content: space-between;
align-content: space-between;
```

Distribuye filas y columnas en el espacio disponible vertical y/o horizontal dejando el mismo espaciado entre ellos pero sin espaciado alrededor



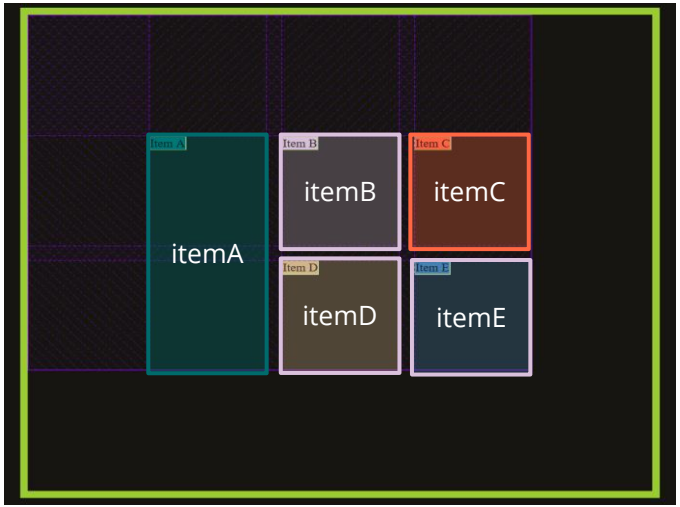
Alineación de elementos en un grid II

```
justify-items || align-items
```

Permiten alinear los elementos contenidos en los grid items (celdas o áreas) de manera simétrica en su espacio horizontal (justify-items) y en el vertical (align-items)

flex-start posiciona los elementos al inicio de su espacio vertical y/o horizontal.

```
justify-items: flex-start;
align-items: flex-start;
```



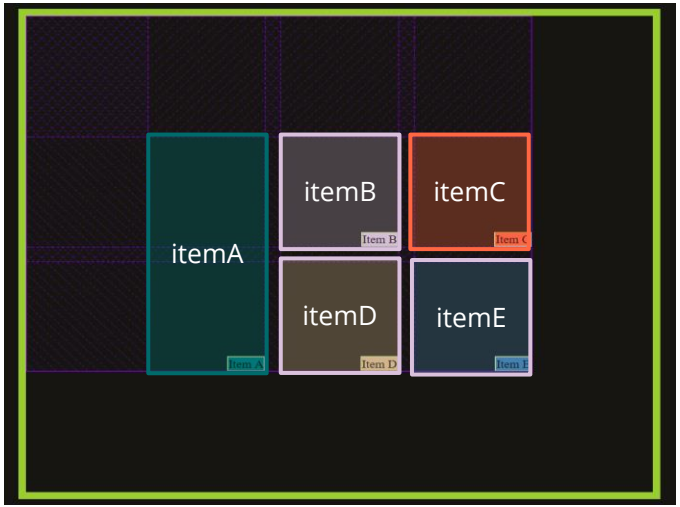
Alineación de elementos en un grid II

```
justify-items || align-items
```

Permiten alinear los elementos contenidos en los grid items de manera simétrica en su espacio horizontal (justify-items) y en el vertical (align-items)

flex-end — posiciona los elementos al final de su espacio vertical y/o horizontal.

```
justify-items: flex-end;
align-items: flex-end;
```



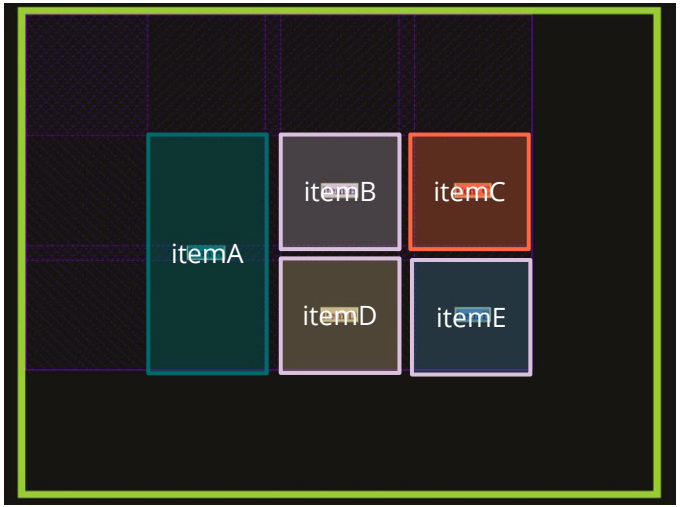
Alineación de elementos en un grid II

```
justify-items || align-items
```

Permiten alinear los elementos contenidos en los grid items de manera simétrica en su espacio horizontal (justify-items) y en el vertical (align-items)

center posiciona los elementos en el centro de su espacio vertical y/o horizontal.

```
justify-items: center;
align-items: center;
```



Alineación de elementos en un grid II

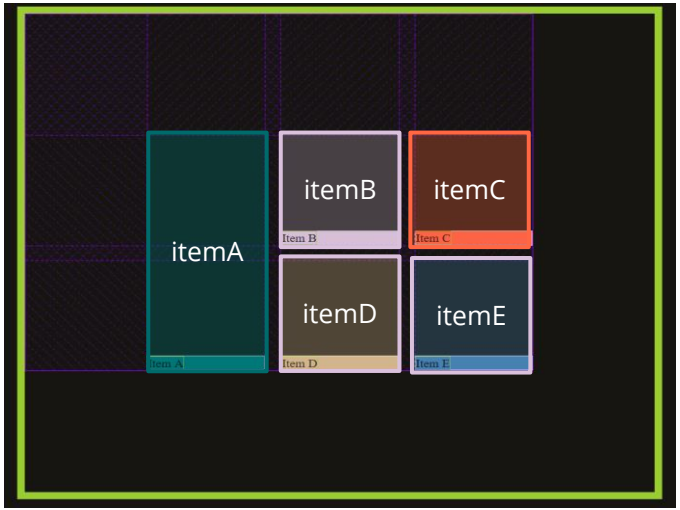
```
justify-items || align-items
```

Permiten alinear los elementos contenidos en los grid items de manera simétrica en su espacio horizontal (justify-items) y en el vertical (align-items)

stretch

```
justify-items: stretch;
align-items: flex-end;
```

Los elementos cubren todo el espacio que tienen disponible verticalmente si la propiedad es align-items y horizontalmente si la propiedad es justify-items.



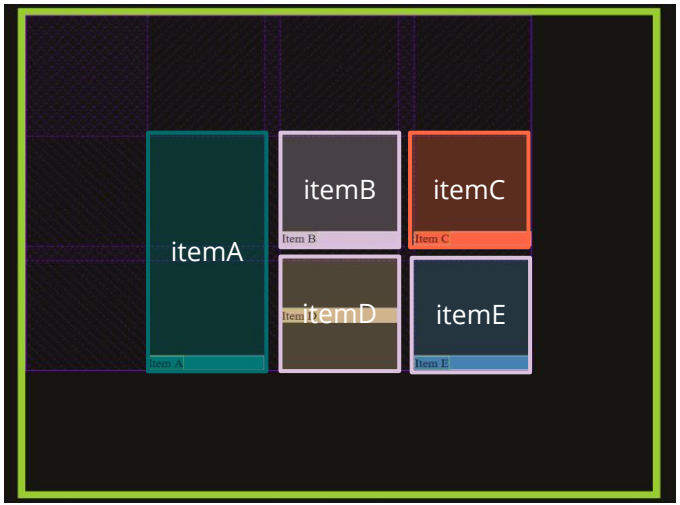
Alineación de elementos en un grid III

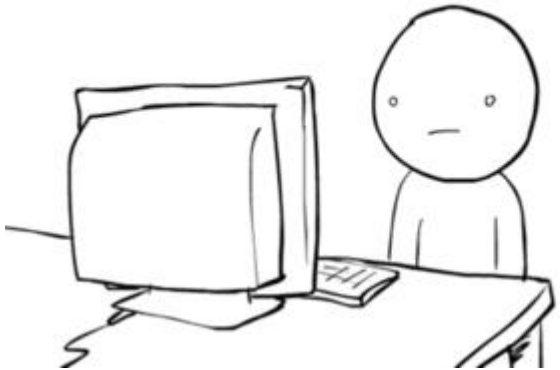
```
justify-selft || align-selft
```

Admiten los mismos valores que **justify-items** y **align-items**. La diferencia es que esta propiedad se utiliza de manera individual en los estilos de cada elemento y no desde el grid contenedor como las anteriores (justify-self para horizontal y align-self para vertical)

```
.grid-container {
  justify-items: stretch;
  align-items: flex-end;
}

.grid-itemD {
  align-self: center;
}
```





Mejor con unos ejemplos
A los teclados!!

¡ Muchas gracias !

@lemoncoders

https://github.com/lemoncode

@basefactorteam